APRENDERAPROGRAMAR.COM

EJEMPLO RECUPERAR DATOS DE FICHEROS JSON EN JAVASCRIPT CON AJAX. JSON.PARSE EVITAR CACHE DE RESPUESTA JSON O XML (CU01215F)

Sección: Cursos
Categoría: Tutorial básico del programador web: Ajax desde cero

Fecha revisión: 2031

Resumen: Entrega nº15 del Tutorial básico “Ajax desde cero”.

Autor: Alex Rodríguez

RECUPERAR INFORMACIÓN JSON CON AJAX

Supongamos que pretendemos recuperar información desde el servidor en segundo plano usando Ajax y que dicha información está en un fichero JSON. El ejemplo que vamos a utilizar es el mismo que hemos estudiado en entregas anteriores del curso, donde hemos visto distintas alternativas para almacenar la información (en un fichero de texto, en un fichero php, en un fichero xml).

DE XML A JSON

Recordemos los datos XML con que habíamos trabajado anteriormente, y veamos su equivalencia en JSON. (El fichero XML y JSON se muestran más abajo).

Los datos consisten en un listado de países, y para cada uno de ellos se dispone información de nombre, capital, un texto descriptivo sobre la capital y una enumeración de los nombres de las ciudades importantes del país.

Nuestro objetivo es que dado un combobox (select html) se recupere la información del servidor y se muestren las ciudades que existan en el archivo de información en función del país que haya seleccionado el usuario.

Ya sabemos cómo recuperar el contenido de un archivo en formato texto. Para ello usamos la propiedad responseText del objeto XMLHttpRequest. Si el texto de respuesta es el contenido de un archivo json, dicho texto contiene la definición de un objeto JavaScript.

Recordatorio:

> En JavaScript los tipos objeto son datos interrelacionados, no necesariamente ordenados, donde existe un nombre de objeto y un conjunto de propiedades y/o métodos. Un objeto puede ser creado específicamente por el programador. No obstante, se dice que todo aquello que no es un tipo primitivo es un objeto y en este sentido también es un objeto, por ejemplo, una función. Consideraremos que las variables son entidades elementales: un número, un carácter, un valor verdadero o falso… mientras que los objetos son entidades complejas que pueden estar formadas por mucha información.
>
> Un objeto en JavaScript puede definirse de varias maneras. Una de ellas es utilizar la notación JSON.

Podemos obtener la cadena de texto que contiene la definición del objeto que vamos a crear simplemente recuperando el contenido del archivo json con esta sintaxis:

```
var jsonResponse = xmlhttp.responseText;
```

Los datos en formato XML que habíamos usado anteriormente y que tomamos como referencia son:

```
<?xml version="1.0" encoding="UTF-8"?>
<listadoPaises>
<pais>
	<nombre>España</nombre>
	<capital>Madrid</capital>
	<textoCapital>Madrid es un municipio y una ciudad de España. La localidad, con categoría…</textoCapital>
	<ciudadImportante>Madrid</ciudadImportante>
	<ciudadImportante>Barcelona</ciudadImportante>
	<ciudadImportante>Valencia</ciudadImportante>
	<ciudadImportante>Sevilla</ciudadImportante>
	<ciudadImportante>Zaragoza</ciudadImportante>
	<ciudadImportante>Málaga</ciudadImportante>
	<ciudadImportante>Murcia</ciudadImportante>
</pais>
<pais>
	<nombre>México</nombre>
	<capital>México D.F.</capital>
	<textoCapital>La Ciudad de México, Distrito Federal o, en su forma abreviada, México, D.F., es… </textoCapital>
	<ciudadImportante>México D.F.</ciudadImportante>
	<ciudadImportante>Ecatepec</ciudadImportante>
	<ciudadImportante>Guadalajara</ciudadImportante>
	<ciudadImportante>Puebla</ciudadImportante>
	<ciudadImportante>Juárez</ciudadImportante>
	<ciudadImportante>Tijuana</ciudadImportante>
	<ciudadImportante>León</ciudadImportante>
	<ciudadImportante>Zapopan</ciudadImportante>
</pais>
<pais>
	<nombre>Argentina</nombre>
	<capital>Buenos Aires</capital>
	<textoCapital>Buenos Aires, formalmente Ciudad Autónoma de Buenos Aires (CABA) —también… </textoCapital>
	<ciudadImportante>Buenos Aires</ciudadImportante>
	<ciudadImportante>Córdoba</ciudadImportante>
	<ciudadImportante>Rosario</ciudadImportante>
	<ciudadImportante>La Plata</ciudadImportante>
	<ciudadImportante>Mar del Plata</ciudadImportante>
	<ciudadImportante>San Miguel de Tucumán</ciudadImportante>
	<ciudadImportante>Ciudad de Salta</ciudadImportante>
</pais>
<pais>
	<nombre>Colombia</nombre>
	<capital>Bogotá</capital>
	<textoCapital>Bogotá, oficialmente Bogotá, Distrito Capital, abreviado Bogotá, D. C. es la… </textoCapital>
	<ciudadImportante>Bogotá</ciudadImportante>
	<ciudadImportante>Medellín</ciudadImportante>
	<ciudadImportante>Cali</ciudadImportante>
	<ciudadImportante>Barranquilla</ciudadImportante>
	<ciudadImportante>Cartagena</ciudadImportante>
	<ciudadImportante>Cúcuta</ciudadImportante>
	<ciudadImportante>Soledad</ciudadImportante>
	<ciudadImportante>Ibagué</ciudadImportante>
</pais>
</listadoPaises>
```

Datos en formato json (archivo listadoPaises.json):

```
{
 "listadoPaises": {
  "pais": [
   {
    "nombre": "España",
    "capital": "Madrid",
    "textoCapital": "Madrid es un municipio y una ciudad de España. La localidad, con categoría…",
    "ciudadImportante": [
     "Madrid",
     "Barcelona",
     "Valencia",
     "Sevilla",
     "Zaragoza",
     "Málaga",
     "Murcia"
    ]
   },
   {
    "nombre": "México",
    "capital": "México D.F.",
    "textoCapital": "La Ciudad de México, Distrito Federal o, en su forma abreviada, México, D.F., es… ",
    "ciudadImportante": [
     "México D.F.",
     "Ecatepec",
     "Guadalajara",
     "Puebla",
     "Juárez",
     "Tijuana",
     "León",
     "Zapopan"
    ]
   },
   {
    "nombre": "Argentina",
    "capital": "Buenos Aires",
    "textoCapital": "Buenos Aires, formalmente Ciudad Autónoma de Buenos Aires (CABA) —también… ",
    "ciudadImportante": [
     "Buenos Aires",
     "Córdoba",
     "Rosario",
     "La Plata",
     "Mar del Plata",
     "San Miguel de Tucumán",
     "Ciudad de Salta"
    ]
   },
   {
    "nombre": "Colombia",
    "capital": "Bogotá",
    "textoCapital": "Bogotá, oficialmente Bogotá, Distrito Capital, abreviado Bogotá, D. C. es la… ",
    "ciudadImportante": [
     "Bogotá",
     "Medellín",
     "Cali",
     "Barranquilla",
     "Cartagena",
     "Cúcuta",
     "Soledad",
     "Ibagué"
    ]
   }
  ]
 }
}
```

Fíjate que el archivo json es más ligero básicamente porque omite la repetición de las etiquetas de que hace uso XML.

CÓDIGO DE EJEMPLO JSON.PARSE

Antes de seguir avanzando crea el archivo json con el contenido indicado anteriormente y súbelo al servidor. Crea también el archivo CU01215F.html con este contenido y prueba los resultados que se obtienen:

```
<!DOCTYPE html><html><head><title>Cursos aprende a programar</title><meta charset="utf-8">
 <style type="text/css">
 *{font-family:sans-serif;} a:link {text-decoration:none;} select{font-size:18px;}
 div div {color: blue; background-color:#F1FEC6; font-size: 20px; float:left; border: solid; margin: 20px; padding:15px;}
 </style>

<script>
function mostrarSugerencia(str) {
var paisElegido='';
if (str=='spain') {paisElegido='España';} else if (str=='mexico') {paisElegido='México';} else if (str=='argentina') {paisElegido='Argentina';}
else if (str=='colombia') {paisElegido='Colombia';} else {paisElegido='none';}

var xmlhttp;
if (str.length==0 ||paisElegido=='none') { document.getElementById("txtInformacion").innerHTML="no hay datos";
mostrarCiudades(); return; }
xmlhttp=new XMLHttpRequest();
xmlhttp.onreadystatechange = function() {
if (xmlhttp.readyState==4 && xmlhttp.status==200) {
                var jsonResponse = xmlhttp.responseText;
                var objeto_json = JSON.parse(jsonResponse);
   paisesRecibidos = objeto_json.listadoPaises.pais;

   for (var i=0; i<paisesRecibidos.length;i++) {
                var nombrePais = objeto_json.listadoPaises.pais[i].nombre;
                                if (nombrePais==paisElegido) {
                                document.getElementById("txtInformacion").innerHTML = 'El pais recibido por get en segundo plano es
'+nombrePais+ ' y tiene indice '+i;
                                var ciudadesPais = objeto_json.listadoPaises.pais[i].ciudadImportante;
                                mostrarCiudades(ciudadesPais);
                                }
                }

}
}

xmlhttp.open("GET","listadoPaises.json?nocache=' + (new Date()).getTime()");
xmlhttp.send();
}

function mostrarCiudades (arrayCiudades) {
var nodoMostrarResultados = document.getElementById('listaCiudades');
if (!arrayCiudades) {nodoMostrarResultados.innerHTML = ''; return}
var contenidosAMostrar = '';
for (var i=0; i<arrayCiudades.length;i++) {
                contenidosAMostrar = contenidosAMostrar+'<div id="ciudades'+i+'">';
                contenidosAMostrar += '<a href="http://aprenderaprogramar.com">' + arrayCiudades[i]+ '</a></div>';
}
if (contenidosAMostrar) {nodoMostrarResultados.innerHTML = contenidosAMostrar;}
}
</script></head>

<body style="margin:20px;"><h2>Elige un país:</h2>
<form action="">
 <select onchange="mostrarSugerencia(this.value)">
  <option value="none">Elige</option>  <option value="spain">España</option>  <option value="mexico">México</option>
  <option value="argentina">Argentina</option>  <option value="colombia">Colombia</option>
</select>
</form>
<br/>
<p>Informacion sobre operacion en segundo plano con Ajax: <span style="color:brown;" id="txtInformacion"></span></p>
<div id="listaCiudades"> </div>
</body></html>
```

Si has seguido los pasos indicados, debes ser capaz de elegir opciones del combobox desplegable y visualizar resultados por pantalla. Por ejemplo, si eliges la opción “Argentina”, el resultado esperado será que por pantalla se visualice:

Informacion sobre operacion en segundo plano con Ajax: El pais recibido por get en segundo plano es argentina y tiene indice 2
Rosario La Plata Mar del Plata San Miguel de Tucumán Ciudad de Salta

Si cambias la selección del país por otro país, debes observar un refresco “casi instantáneo” de modo que se modifica el texto y las ciudades que se muestran, viéndose en cada caso las correspondientes al país seleccionado.

Cada vez que seleccionamos un país, hay una comunicación con el servidor en segundo plano y la información recibida del servidor la utilizamos para hacer cambios en la página web sin necesidad de hacer una recarga tradicional.

Este resultado es idéntico al que obtuvimos en una entrega anterior del curso. La diferencia está en que ahora estamos recuperando los datos de servidor desde un archivo JSON y trabajando con este formato de datos. La ventaja de usar JSON en lugar de recuperar datos desde un archivo PHP es que nos independizamos del lenguaje. La ventaja frente a usar XML es que JSON es más ligero y rápido.

TRANSFORMAR LA RESPUESTA EN UN OBJETO JAVASCRIPT

A partir del contenido del archivo json podemos generar un objeto JavaScript. Una vez generemos el objeto JavaScript ya podremos manipularlo de forma cómoda con notación de punto y notación de array.

JSON.parse utiliza un objeto predefinido JavaScript (el objeto JSON) para convertir la cadena de texto con la definición del objeto en un objeto en sí.

Hay una alternativa que ha sido comúnmente usada para generar el objeto JavaScript antes de la existencia del objeto JSON. Dicha alternativa se basa en el uso de eval y su sintaxis sería la siguiente:

```
var objeto_json = eval("("+jsonResponse+")");
```

eval ('cadenaDeTextoPasadaALaFuncion'); tiene dos implicaciones: una evaluación o ejecución, y la obtención de un valor de retorno (si existe). En este caso el valor de retorno es un objeto según la definición contenida en la cadena de texto.

El uso de eval está desaconsejado ya que implica riesgos de seguridad. Se recomienda usar JSON.parse.

ANÁLISIS DEL CÓDIGO

Centraremos nuestra atención en este fragmento de código:

```
var xmlhttp;
if (str.length==0 ||paisElegido=='none') { document.getElementById("txtInformacion").innerHTML="no hay datos";
mostrarCiudades(); return; }

xmlhttp=new XMLHttpRequest();
xmlhttp.onreadystatechange = function() {
if (xmlhttp.readyState==4 && xmlhttp.status==200) {
        var jsonResponse = xmlhttp.responseText;
        var objeto_json = JSON.parse(jsonResponse);
  paisesRecibidos = objeto_json.listadoPaises.pais;

  for (var i=0; i<paisesRecibidos.length;i++) {
        var nombrePais = objeto_json.listadoPaises.pais[i].nombre;
                if (nombrePais==paisElegido) {
                document.getElementById("txtInformacion").innerHTML = 'El pais recibido por get en segundo plano es
'+nombrePais+ ' y tiene indice '+i;
                var ciudadesPais = objeto_json.listadoPaises.pais[i].ciudadImportante;
                mostrarCiudades(ciudadesPais);
                }
        }

}
}
xmlhttp.open("GET","listadoPaises.json?nocache=' + (new Date()).getTime()");
xmlhttp.send();
```

Una vez recuperamos el contenido del archivo json en la línea var jsonResponse = xmlhttp.responseText; lo transformamos en un objeto usando:

var objeto_json = JSON.parse(jsonResponse);

Una vez tenemos el objeto ya podemos acceder a sus propiedades con notación de punto y/o notación de array. Por ejemplo podríamos obtener el nombre del primer país de la lista de países escribiendo var nombrePrimerPais = objeto_json.listadoPaises.pais[0].nombre;

La propiedad listadoPaises consta de una serie de elementos a los que podemos acceder por índice. Para acceder al primer elemento usaríamos el índice cero. Cada elemento país está compuesto por nombre, capital, textoCapital y un array de ciudades importantes denominado ciudadImportante. Para acceder al nombre del primer país (España) escribimos objeto_json.listadoPaises.pais[0].nombre. Si quisiéramos acceder a la tercera ciudad del primer país escribiríamos:

objeto_json.listadoPaises.pais[0].ciudadImportante[2]

Una vez tenemos el objeto con la definición del archivo json extraemos otro objeto asemejable a un array que contiene la información de los países.

paisesRecibidos = objeto_json.listadoPaises.pais;

Este nuevo objeto lo recorremos con un bucle for extrayendo la propiedad nombre de cada elemento recorrido: var nombrePais = objeto_json.listadoPaises.pais[i].nombre;

Cuando comprobamos que el nombre del elemento coincide con el nombre buscado, extraemos el valor de ciudadImportante, que es un array de elementos, mediante la instrucción:

var ciudadesPais = objeto_json.listadoPaises.pais[i].ciudadImportante;

Una vez tenemos el array ciudadesPais simplemente tenemos que recorrerlo para mostrar la información por pantalla.

EVITAR EL CACHEADO DE RESPUESTAS

Los datos recuperados usando GET pueden quedar cacheados en los navegadores. Esto significa que si repetimos una petición para recuperar datos con GET que ha sido realizada anteriormente, podemos obtener los mismos resultados en el navegador incluso si los datos han cambiado en el servidor (debido al cacheado en el navegador).

Para evitar el cacheado de respuesta hemos usado esta expresión:

xmlhttp.open("GET","listadoPaises.json?nocache=' + (new Date()).getTime()");

Aquí la url invocada es una url a la que se añade una terminación aleatoria irrelevante para el resultado pero que da lugar a que el navegador interprete la petición como si se tratara de una nueva petición evitando el cacheado.

Si se realizan las peticiones con POST no ocurre este problema.

EJERCICIO

Modifica el código que hemos usado como ejemplo para realizar la petición de datos con ajax mediante POST en lugar de mediante GET.

Para comprobar si tus respuestas son correctas puedes consultar en los foros aprenderaprogramar.com.

Próxima entrega: CU01216F

Acceso al curso completo en aprenderaprogramar.com -- > Cursos, o en la dirección siguiente: http://aprenderaprogramar.com/index.php?option=com_content&view=category&id=83&Itemid=212